УДК 681.3.068
А. А. Бурыкин, Е. Н. Конюков, В. Ю. Генералов, Т. В. Вишнева
ФГАОУ ВО «Уральский федеральный университет имени первого Президента России Б.Н. Ельцина», г. Екатеринбург, Россия

ХРАНЕНИЕ ОБЪЕКТОВ СЕССИИ С УЧЕТОМ РАЗДЕЛЕНИЯ ВКЛАДОК БРАУЗЕРА НА ПРИМЕРЕ ИСТОРИИ ПЕРЕХОДОВ ПОЛЬЗОВАТЕЛЯ НА ПЛАТФОРМЕ .NET MVC

Аннотация

Статья посвящена разработке технологии хранения истории переходов пользователей с учетом разделения вкладок браузера на платформе .NET MVC. В статье рассмотрена актуальность данной технологии. Приведено описание используемых средств разработки.

Ключевые слова: сессия, web, ссылка, .NET объект, MVC.

Abstract

This article was presented development of technology for storing user requests into server session with browser's tab separation for .NET MVC. We considered the relevance of technology. The description of used software for development was reviewed in this article.

Keywords: session, web, reference, .NET object, MVC.

Становится сложно игнорировать растущую популярность web систем. Все большее предпочтение среди корпоративных систем отдается именно web среде. Однако, пользователь, зачастую привыкший к оконному пользовательскому интерфейсу, к операционной системе windows, часто сталкивается с рядом трудностей. Необходимо понимать, что система, которая взаимодействует по http протоколу, обладает рядом особенностей, накладывающих ряд ограничений на поведение системы. Так, если пользователь запустит несколько windows приложений локально и будет в них одновременно работать, то приложения в большинстве случаев не будут между собой коррелировать, продолжая независимо работать. В то время как, если пользователь откроет несколько вкладок браузера одной и той же корпоративной системы, (поскольку оболочка у вкладок общая – браузер), то все вкладки будут работать в едином контексте. На первый взгляд в этом нет никаких проблем. Однако на практике это приводит часто к неожиданному с точки зрения пользователя поведению.

Разрабатывая под web платформу, разработчик вынужден использовать механизмы, которые предоставляет браузер для взаимодействия с пользователем, в том числе для хранения информации на стороне пользователя (cookie файлы) и на стороне сервера (текущая сессия).

Современные браузеры обладают одинаковым перечнем встроенных функций, среди которых кнопки – обновить страницу, вперед и назад. Встроенного функционала кнопки «Назад» часто бывает недостаточно. Рассмотрим типичный пример: пользователь перешел из списочной формы объектов в краткую форму просмотра объекта, далее – в расширенную форму. После чего ему необходимо перейти назад к списочной форме, очевидно, что кнопка «Назад» должна вести именно туда (рисунок 1). Либо пользователь изменил данные на расширенной форме объекта, нажал «Применить» (отправлен Post запрос с данными) и далее по кнопке «Назад» ожидает переход на предыдущую форму, а именно – списочную форму объектов (рисунок 2). На рисунках – желтыми стрелками обозначены переходы по встроенной кнопке «Назад», зелеными – логичные, ожидаемые пользователем.

Данная задача решается достаточно тривиально: необходимо хранить в сессии список переходов пользователя и номер текущего перехода. Далее при нажатии кнопки «Назад» на форме (либо при других действиях пользователя) возвращаться к предыдущим формам. Это достигается за счет извлечения нужного перехода из сессии.

Session[“req_1”] = “http://...”;

```
Session[“req_2”] = “http://...”;
…
Session[“req_n”] = “http://...”;
Session[“req_counter”]=n;
```

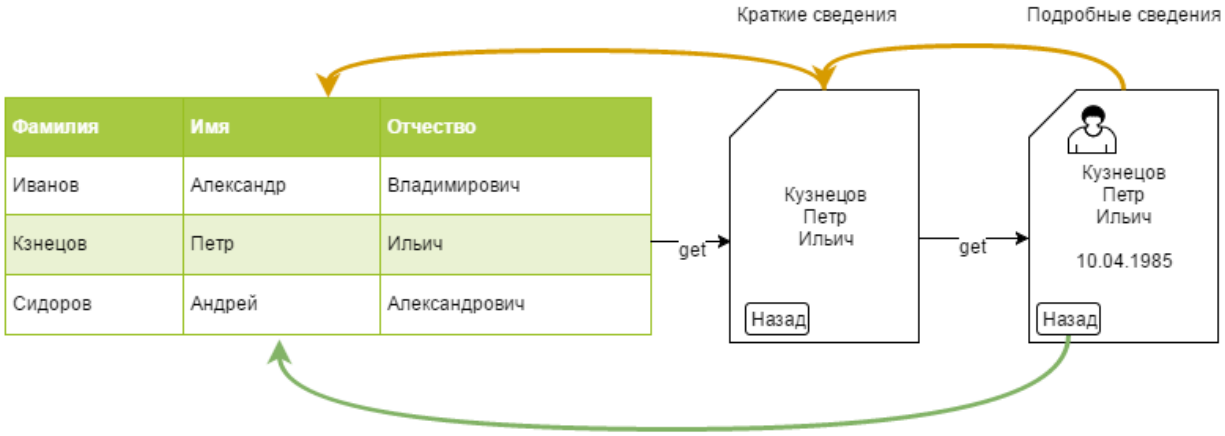

Рис. 1. Пример работы с краткой/расширенной формой

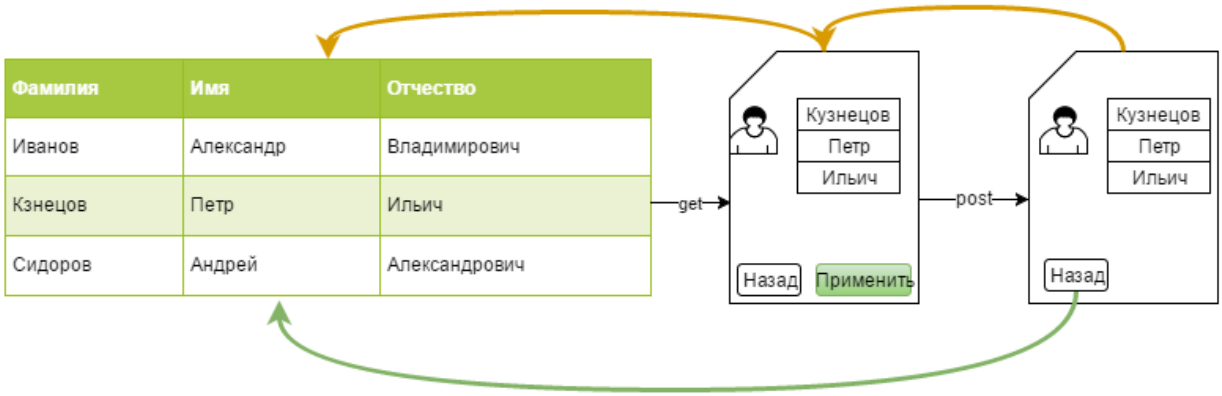

Рис. 2. Пример работы с отправкой данных на сервер

Здесь следует понимать, что пользователь может достаточно долго находиться в одной сессии и потенциально это ведет к утечке памяти, поэтому размер стека переходов необходимо ограничить, например, N переходами. Таким образом, если номер текущей записи (n) достигает N, то запись n–N (если она существует) необходимо удалять.

Данный механизм неплохо работает и показывает свою эффективность. Однако, у такого подхода есть один существенный недостаток. При работе пользователя в нескольких вкладках браузера возникает ситуация, когда пользователь, нажимая «Назад» на одной вкладке, переходит к url-запросу из соседней вкладки. Данное поведение логично с точки зрения разработчика – один пользователь, одна сессия, одна история запросов, но не с точки зрения пользователя!

Давайте попробуем, усложнить существующий механизм так, чтобы история хранилась с учетом разделения по вкладкам. С точки зрения лишь сервера данную задачу решить невозможно – браузер нигде не включает информацию о том, с какой вкладки сейчас приходит запрос, поэтому данный функционал должен быть частью самой системы. Необходимо при каждом запросе пользователя указывать номер вкладки, с которой пришел запрос.

Рассмотрим, как формируется запрос в парадигме MVC .Net:

http://сервер/контроллер/метод/параметры

Указанное поведение является поведением по умолчанию и рекомендуемым. Тем не менее, никто не запрещает указанное поведение немного изменить – добавить в запрос номер вкладки:

http://сервер/вкладка/контроллер/метод/параметры

Для хранения на сервере следует немного модифицировать существующий механизм – добавить к ключу сессии номер вкладки t из get запроса. Максимальное число вкладок также следует ограничить – T.

Session[“req_t_1”] = “http://...”;
Session[“req_t_2”] = “http://...”;
…
Session[“req_t_n”] = “http://...”;
Session[“req_t_counter”]=n;

Рассчитаем максимальный размер истории:

Максимальный размер get-запроса: 2083 байт, T = 10 вкладок, N = 20 запросов.

2083 * 10 * 20 = 416600 байт

Если рассматривать среднюю корпоративную систему, с которой одновременно работает около 100 сотрудников, то указанный подход при самом неблагоприятном исходе займет не более 50 мегабайт памяти.

Очевидно, что данный подход требует дополнительных вычислений и оперативной памяти, однако позволяет сделать поведение системы удобным и предсказуемым.

Список использованных источников

1. Частые вопросы [сайт]. URL: http://vpros.ru/34-kakaya-maksimalnaya-dlina-url-adresa-v-raznykh-brauz

2. CYBERGURU.ru [сайт]. URL: http://www.cyberguru.ru/microsoft-net/asp-net/aspnet-session-exploration.html

УДК 658.512.6

А. С. Бычкова, А. Б. Нечаева
ФГБОУ ВО «Орловский государственный университет имени И.С. Тургенева»,
г. Орел, Россия

КОНСТРУКТИВНЫЕ ОСОБЕННОСТИ РАЗРАБОТКИ СЕРВИСА АВТОМАТИЗАЦИИ СОСТАВЛЕНИЯ ПРОГРАММ ТРЕНИРОВОК С УЧЕТОМ ФИЗИОЛОГИЧЕСКИХ ОСОБЕННОСТЕЙ ПОЛЬЗОВАТЕЛЯ

Аннотация

В данной статье была обоснована актуальность создания сервиса автоматизации составления программ тренировок с учетом физиологических особенностей пользователя. Была показана категории лиц, которым может быть полезен сервис. Были обоснованы причины разработки мобильного приложения сервиса. Описывались конструктивные особенности разработки сервиса автоматизации. Формировались ограничения, накладываемые на архитектуру и структуру проектируемого сервиса, который будет представлять собой распределенный программный комплекс. Аргументировался перенос части функций сервиса на мобильное приложение. Доказывалась необходимость синхронизации данных между базами данных мобильного приложения и сервера. Показывалась имеющаяся в сервисе система подчинения и тесная взаимосвязь между различными подсистемами сервиса. Аргументировалась необходимость доступа к платным функциям сервиса с мобильного телефона только в онлайн режиме. Приводилась логическая схема построения сети, система подчинения подсистем сервиса, структурная схема сервиса, инфологическая модель базы данных.